Haaga-Helia
University of Applied Sciences

Master's Thesis

Test Automation Process Improvement

A case study of BroadSoft

Jalendar Gummadi

Master's Thesis
Degree Programme in Information Systems Management
2016

Haaga-Helia
University of Applied Sciences

Abstract

Date

Author(s)
Jalendar Gummadi

Degree programme
Information Systems Management, Master's Degree

Report/thesis title
Test Automation Process Improvement
A case study of BroadSoft

Number of pages and attachment pages
41 + 4

This master thesis research is about improvement of test automation process at BroadSoft Finland as a case study.

Test automation project recently started at BroadSoft but the project is not properly integrated in to existing process. Project is about converting manual test cases to automation test cases. The aim of this thesis is about studying existing BroadSoft test process and studying different test automation frameworks. In this thesis different test automation process are studied and choosing of right automation framework for the project.

The other aim is to improve regression test process. So this thesis involves studying of current automation process and different tools which can be used for improving regression process.

This thesis will present improvement ideas for test automation process includes process level improvements, framework level improvements and regression testing level improvements.

Keywords
Test Automation Process, Test automation Frameworks, Regression Improvements, Process Improvements, Case study

Table of contents

1 Introduction 1
1.1 Background 1
1.2 Aim and Objectives 2
1.3 Research questions 3
1.4 Scope 3
1.5 Motivation 4
1.6 Structure of report 4
2 Research Methodology 6
2.1 Action research 6
2.2 Data gathering methods 8
2.2.1 Data collection 8
2.2.2 Interviews or Discussions. 8
3 ITIL Service Transition 9
3.1 Information technology infrastructure library (ITIL) 9
3.2 ITIL service transition in BroadSoft test automation 11
3.2.1 Change Management 11
3.2.2 Release and Deployment Management 12
3.2.3 Service Validation and Testing 12
3.2.4 Knowledge Management 12
4 Test process 14
4.1 Manual test process 14
4.1.1 BroadSoft manual test process 15
4.2 Automation Test Process 16
4.2.1 BroadSoft Automation test process 16
4.3 Robot Framework 18
4.3.1 Keyword-driven/Table-driven testing 18
4.4 Robot Test Cases 18
4.4.1 Writing Test Cases 19
4.4.2 Managing Test Cases 19
4.4.3 Running Test Cases 19
4.5 Current BroadSoft regression test process 19
4.5.1 Regression Testing 20
4.5.2 Regression testing test data 20
5 Test Automation Frameworks and improvement ideas 21
5.1 Types of Test Automation Framework 21
5.1.1 Module Based Testing Framework 22
5.1.2 Library Architecture Testing Framework 23

5.1.3 Data Driven Testing Framework....24
5.1.4 Keyword Driven Testing Framework....25
5.1.5 Hybrid Testing Framework....27
5.1.6 Behaviour Driven Development Framework....28
5.2 Key Driven Test Process and Improvement ideas....30
5.2.1 Process....30
5.2.2 Maintenance....31
5.2.3 Expertise....31
5.2.4 Keywords....31
6 Regression testing and improvement ideas....33
6.1 Regression test strategy....34
6.2 Kalistick Test Scoring tool and results....35
6.2.1 Links between changes and tests....35
6.3 Results....36
6.4 Regression improvements....36
6.4.1 Test Link plugin....36
7 Recommendation and development ideas....37
References....39
Attachments....42
Attachment 1. Testlink for creating manual test-cases....42
Attachment 2. Jira Tool used for tracking whole product process....42
Attachment 3. Screenshot of Jenkins build job....43
Attachment 4. Screenshot shows summary of test-cases from Jenkins....43
Attachment 5. Screenshot shows summary of robot test cases (Pass/Failure)....44
Attachment 6. Screenshot show integration of testlink in Jenkins job....44

Abbreviations

QA	Quality Assurance
VEA	Very Early Adapter
EA	Early Adapter
RC	Release Candidate
ITIL	Information Technology Infrastructure Library
AR	Action Research
POC	Proof of Concept
KDT	Key Driven Testing
SW	Software
AUT	Application under Test
ATDD	Acceptance test driven development
TSV	Table Separated Value
HTML	Hypertext Mark-up Language
CCTA	Central Computing and Telecommunication Agency
POC	Proof of Concept
TSV	Tab Separated Values
OOP	Object Oriented Programming

1 Introduction

The main aim of this project is to Improve test automation process at BroadSoft Finland division. From many years this division is working on manual test process. Recently this division started test automation but test automation is not properly integrated in current process. There are many advantages of test automation like it reduces manual work and to improve product quality (Advantages of Automation, 2016).

This project is done at BroadSoft Finland. BroadSoft is a global communication software provider with several deployments worldwide supporting millions of subscribers. BroadSoft Finland works on software development for desktop client application. This division wants to have test automation to its product, till now most of the test process is done using manual testing.

To implement test automation we have chosen some tools like Robot Framework (Robot, 2016), Squish (Froglogic, 2016) for writing automation test cases. We have started developing some automatic test cases but these automatic test cases are not became part of process. The plan is to have test automation as part of the current test process to bring effective results.

This study focuses on improving automation test process at BroadSoft Finland. This research contains both theoretical and practical parts. In theoretical part studied different test automation frameworks and also studied ways to improve regression testing process. From studies chosen one test automation framework to fit in current process and also chosen the ways to improve regression testing process. For improving test process there is need of understanding current test process, so this research also includes studying of current manual and automation test process. In this thesis the term ‘BroadSoft’ is used frequently, it should be understood as ‘BroadSoft Finland’.

1.1 Background

BroadSoft Finland division works on Software Development. This division develops and maintains Software and this division is also responsible of testing the product .The QA (Quality Assurance) team is responsible in testing the product. QA team uses manual test process for testing product. Manual test process includes writing test-cases, executing test-cases, maintaining test-case and running regression test-cases.

This division recently started working on test automation project. In test automation process we are using tools such as squish, robot framework and Jenkins. These tools are used for writing test cases and running continuous integration jobs. We wrote some of automate test cases but test automation is not properly integrated to current process. Even though we have automated test case we used to depend on manual testing process. From here the idea of this thesis is started how to integrate test automation in to current process and improve test automation process.

We used to release different versions of product in each year. So for every release we used to run all test cases called ‘regression test plan’. At present we have more than 1500 test cases which we were run for each release. The regression testing takes around 3 to 4 weeks to complete full regression test plan for each release. The test automation improves regression testing time period. This is also one of criteria of this research to use test automation effectively in regression testing.

1.2 Aim and Objectives

The aim of this project is to have test automation which involves converting manual testing to automatic testing. Automatic testing will be done to existing test cases and also to upcoming new feature test cases. In general test automation benefits is to save time, speed, and repeatability, reusable and to reduce cost.

The objectives of this research involves in studying different test automation frameworks and selecting best test automation framework which will fit into our current process.

Some of other objectives is to have automation testing at different levels of current process. The research framework should help in bringing automation testing at different level of the process. It should also help in writing automated test-cases, executing test-cases, maintain test-cases and also should help in release management.

The current test process involves writing manual test-cases and executing these test-cases. We have more than 1500 test-cases and most of these test-cases will be executed during each release. Execution time will takes around 3 to 4 weeks to run full release testing. The objective of this project involves is to reduce the execution time.

Out of all test-case only some test cases can be automated using automation tools due to some technical reasons. So in the initial phase we have chosen to automate sanity test-cases, these test-cases were automated but not effectively used in releasing the product.

We still used to do manual testing even we already have automation test cases. In future we will have some more automation test-case. The aim of this project to eliminate the manual work if test automation is available. The other objective is to bring test automation in release management process in bringing effective results.

So the overall research of this thesis involves:

- Choosing a test automation framework which will fit test automation in different levels of current test process.
- Reduce release/regression testing time period.
- Using test automation in release management process.

1.3 Research questions

From the above aim and objectives we (after reviewing meeting) have framed some of the research question.

Main research questions and corresponding sub-section are listed below:

How to measure the overall effectiveness of test automation?
How to reduce release or regression testing time period?
How effectively use test automation in release management?

How to apply test automation at different levels of process?
How test process is currently followed at BroadSoft?
How to choose a right test automation framework?

1.4 Scope

BroadSoft as a company operates in different business units depending of nature of offered services. There are many business units but this research will be focused on desktop application.

The author is working in business unit and aware of corresponding business. This research is to study a test automation frameworks which will fit test automation into current process. This research doesn't include changing current process. This research doesn't include any training process. The test automation will done using Robot framework, Squish tools so this

research will focused on around these tools. The test automation also involves documentation, deployment which is out-scope of this research since author is not involved in process.

1.5 Motivation

The motivation of this research was in both company and personal level. In effective use test automation there is need of proper test automation framework, which brings test automation into current process and then we can use effectively test automation. Since I (Author) am working in Broad-Soft as Senior QA (Quality Assurance) engineer, I having been involved in manual testing. The manual testing involves more repetitive work. The test automation project is to automate repetitive work. I have been involved in test automation project. My role will be to integrate this test automation to current process. The integration is to have test automation at different levels of the test process.

1.6 Structure of report

This thesis is divided in to seven chapters, details of each chapter are listed below.

Chapter 1 presents details about project background and its aim and objectives. This chapter also give project research questions and its scope.

Chapter 2 and Chapter 3 presents research methodology and ITIL service transition pertaining to this thesis. Chapter 3 gives details about ITIL service transition in theory and its mapping to BroadSoft test automation project.

Chapter 4 presents current test process in BroadSoft. This chapter will present current manual and automation test process at BroadSoft and also gives current regression test process at BroadSoft.

Chapter 5 presents study of different test automation frameworks and the frameworks chosen for existing test automation project. This chapter also give Improvement ideas to improve test automation process.

Chapter 6 presents regression test process improvement ideas and different tools which can be used to improve regression testing.

Chapter 7 presents conclusions about test automation project improvements in process level, test automation framework improvement ideas, regression test improvement ideas and future research ideas.

2 Research Methodology

2.1 Action research

This research and the project utilized was Action Research (AR) method. The research is to improve test automation process so the action research is the best methodology for this research.

According to Meyer (2000) “Action research strength lies in its focus on generating solutions to practical problems and its ability to empower practitioners, by getting them to engage with research and the subsequent development or implementation activities”.

Action research works through a cyclical four steps process of consciously and deliberately:

1. Planning
2. Taking action
3. Evaluating the action
4. Leading to further planning and so on

(McDermott, Aoife Mary, D. Coghlan, 2008) Action research is participatory in nature, which led (Kemmis and McTaggart, 2000) to describe it as participatory research. Figure 1 illustrates the spiral model of action research proposed by (Kemmis and McTaggart 2000), although the author do not recommend that this is used as rigid structure.

Below in Figure 1 shows how (Kemmis and McTaggart’s) AR spiral process

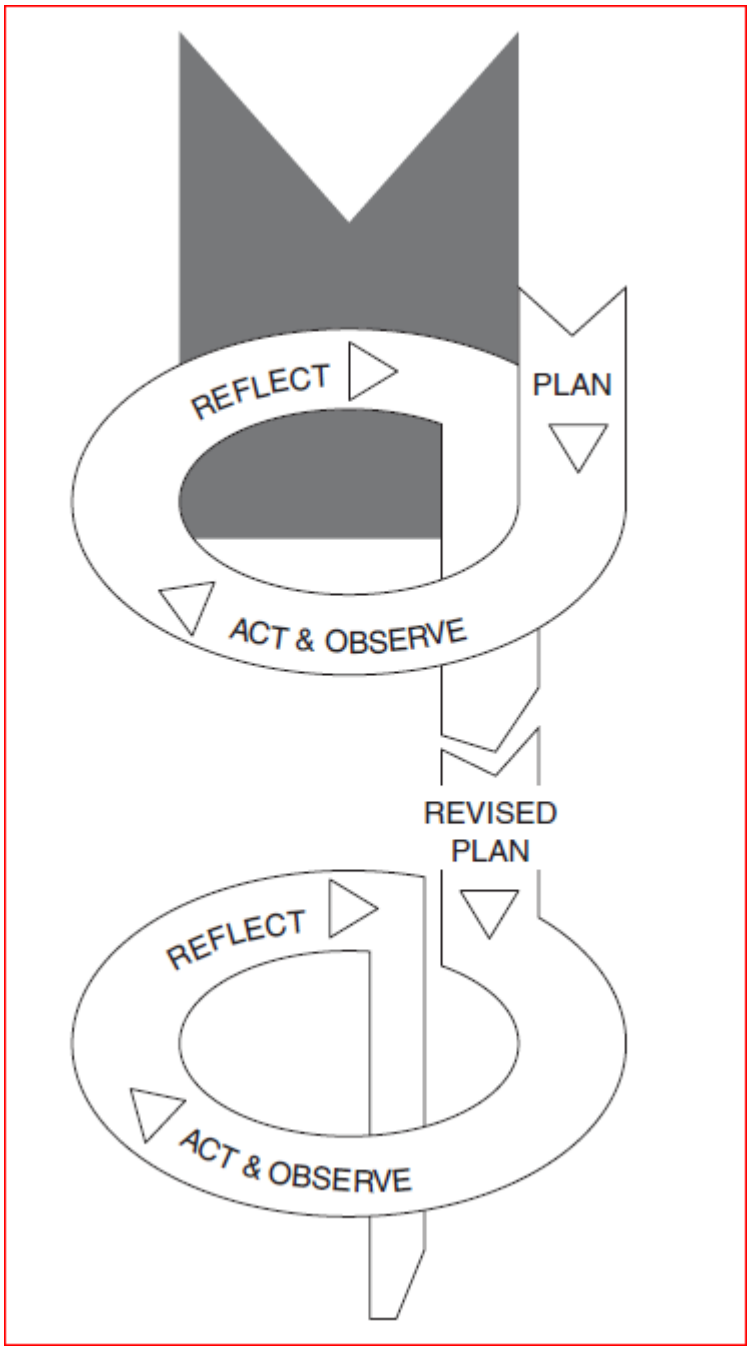


Figure 1. Kemmis and McTaggart's action research spiral

Participatory action research is a learning process whose fruits are the real and material changes in the following (Kemmis and McTaggart's, 1998):

- What people do?
- How people interact with the world and with others?
- What people mean and what they value?
- The discourses in which people understand and interpret their world?

2.2 Data gathering methods

This work is on improving test automation process and I am part of current test automation process. I took internal existing data which is used for improving test automation process and Informal team interviews/discussions and their views in general test automation. Mainly my views and some team views collectively will reflect as a part of this thesis documents.

2.2.1 Data collection

I need data such has how many regression test cycles and how many test cases run and how releases done, so I took for manually all the data from internal test-link tool. This data is mainly used as basis for improving regression testing.

2.2.2 Interviews or Discussions.

In this research I had many times Informal interviews/discussions with my colleagues who are working test automation and used to discuss on test automation improvements on frameworks and process level. This project has limited resource when the project got started and I am part of this project, so there is no need separate questionnaires. Since the discussion were Informal so I have reflected their views and mainly my view as part of this thesis. The discussion made effective utilization of action research method. These discussion/interviews helped in improving test automation process in general and selecting best framework which will fit into current process.

3 ITIL Service Transition

This section is about ITIL and ITIL service Transition. The ITIL Service Transition studied as part of this research because BroadSoft Finland division mainly works on ITIL service transition state. This division works on software development and software maintenance to its product. Test process improvement is part of ITIL Service Transition area. First let's discuss some of theoretical part of ITIL and ITIL service transition in theory and BroadSoft point view.

3.1 Information technology infrastructure library (ITIL)

The Information Technology Infrastructure Library (ITIL) is a collection of concepts and practices combined in a series of books to be applied in IT Service Management (ITSM), IT development and IT operations. At first it was created in late of 1980 by Central Computing and Telecommunication Agency (CCTA). The names ITIL and IT Infrastructure Library are registered trademarks of the United Kingdom's Office of Government Commerce (OGC) (ITIL v3, 2014).

The ITIL 2011 was used in this study, as main framework. ITIL 2011 consists of five lifecycle publications and each publication addresses capabilities having direct impact on a service provider's performance. The ITIL 2011 publications are:

- ITIL Service Strategy
- ITIL Service Design
- ITIL Service Transition
- ITIL Service Operation
- ITIL Continual Service Improvement

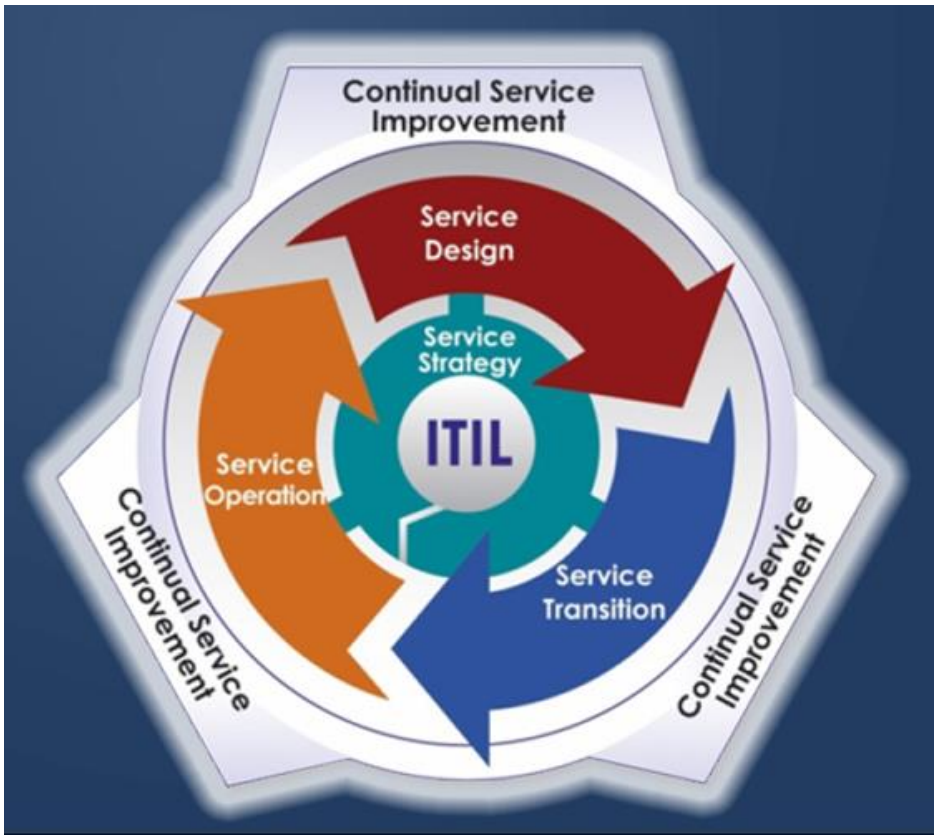


Figure 2. ITIL Service Lifecycles (ITIL V3 Application Support, 2008)

The above Figure 2 shows all services life cycles and how they are related to each other. ITIL service strategy is the core of ITIL which will serve as a strategy to serve customers. The service strategy determines which services IT organization offers. The service strategy is the input for service design which produces new IT services. The new services are taken in to use in service transition life cycle. The service transition is the area we will focus, since BroadSoft Finland divisions works on this life cycle.

ITIL is organized around a service lifecycle which includes service strategy, service design, service transition, service operation and continual service improvement (ITIL V3 vs ITIL 2011)

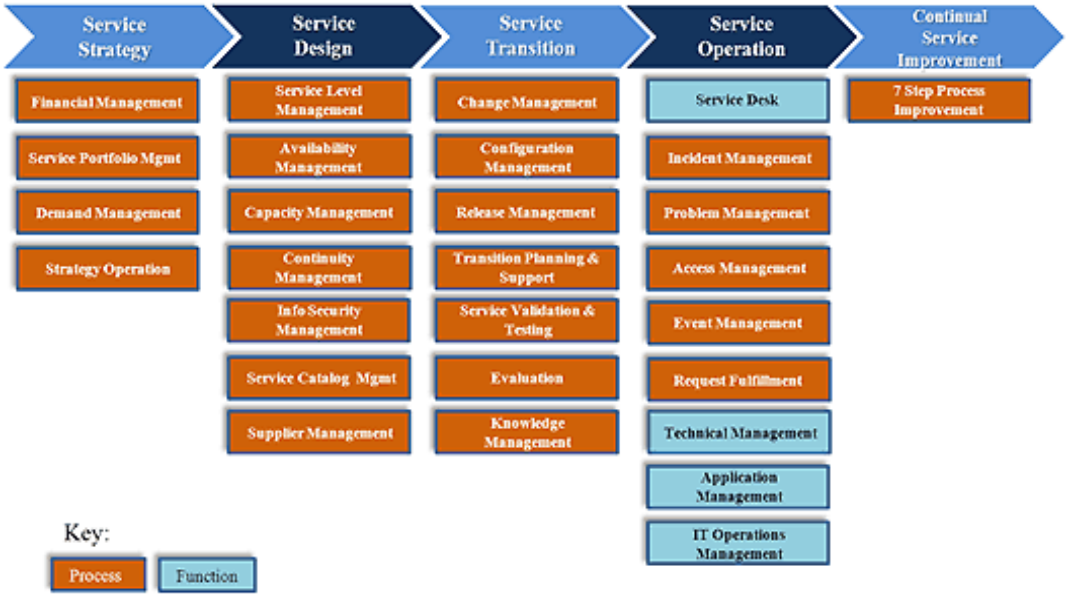


Figure 3. ITIL Process (ITIL V3 vs ITIL 2011)

3.2 ITIL service transition in BroadSoft test automation

ITIL service transition helps plan and manage the change of state of a service in its lifecycle. Managing risk for new, changed and retired services protects the product environment. This helps the business deliver value to itself and its customers (ITIL Service Transition). Service Transitions includes these phases(ITIL V3 vs ITIL 2011):

1. Change management
2. Change Evaluation
3. Project Management (Transition planning and support)
4. Application Development
5. Release and Deployment Management
6. Service Validation and Testing
7. Service Asset and Configuration Management
8. Knowledge Management

For this test automation improvement research we are interested in change management, release and deployment management, service validation and testing and knowledge management.

3.2.1 Change Management

Process Objective:

To control the lifecycle of all Changes. The primary objective of Change Management is to enable beneficial Changes to be made, with minimum disruption to IT services (ITIL V3 vs ITIL 2011).

Test Automation project view:

Test automation is a new change so change management is needed in current test automation project. Test automation as change should not affect in existing functionality or process of the product.

3.2.2 Release and Deployment Management

Process Objective:

To plan, schedule and control the movement of releases to test and live environments. The primary goal of Release Management is to ensure that the integrity of the live environment is protected and that the correct components are released (ITIL V3 vs ITIL 2011).

Test Automation project view:

Test automation makes regression testing in different way than usual manual testing this test automation helps release process and helps in fastening regression testing. But project need to make sure test automation should affect existing release process.

3.2.3 Service Validation and Testing

Process Objective:

To ensure that deployed Releases and the resulting services meet customer expectations, and to verify that IT operations is able to support the new service (ITIL V3 vs ITIL 2011).

Test Automation project view:

With test automation project need take care of existing customer expectation. The project also shouldn't break IT operations who supports new service to our product.

3.2.4 Knowledge Management

Process Objective:

To gather, analyze, store and share knowledge and information within an organization. The primary purpose of Knowledge Management is to improve efficiency by reducing the need to rediscover knowledge (ITIL V3 vs ITIL 2011).

Test Automation project view:

The test automation is new project so the knowledge need to documented & stored and this knowledge should be shared. There proper knowledge management so that new person should rediscover knowledge.

4 Test process

To improve test automation process, it is important to study existing BroadSoft test processes (Manual and Automation). This section will give in details of current manual test process and automation test process at BroadSoft. This project most effectively utilized action research methodology, so I will also give in details how test automation process improved from the day it started to present situation. This section will also cover both manual and automation test process in theory and their corresponding mapping with existing BroadSoft manual test process and test automation project.

4.1 Manual test process

Testing is a process rather than a single activity (Istqbexamcertification.com, 2016). This process starts from test planning then designing test cases, preparing for execution and evaluating status till the test closure. So, we can divide the activities within the fundamental test process into the following basic steps (Istqbexamcertification.com, 2016).

The Fundamental Test Process comprises five activities:

- Planning & Control
- Analysis & Design
- Implementation & Execution
- Evaluation Exit Criteria and Reporting
- Test Closure Activities

Below Figure 4, shows how general manual test process goes from planning to test closure.

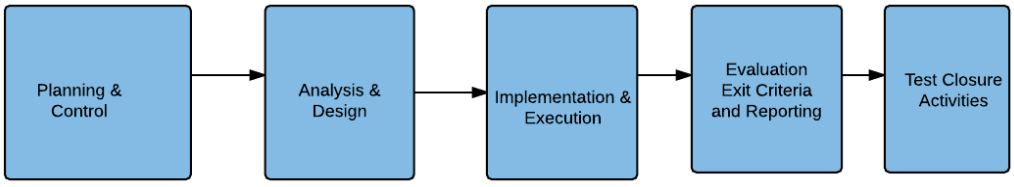


Figure 4. Manual test process, Adapted from Istqbexamcertification.com, 2016.

4.1.1 BroadSoft manual test process

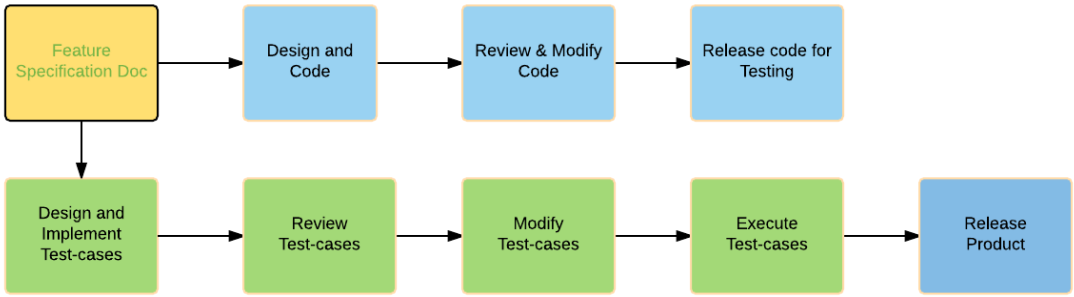


Figure 5. Manual test process at BroadSoft.

The current BroadSoft manual test process goes through these phases:

1. Design and Implementation of test-cases
2. Review test cases
3. Modify test cases based on review comments
4. Execute test-cases.

Manual test-cases are written using testlink tool (Attachment 1). Attachment 1 available at end of this thesis shows how manual test-cases are maintained in testlink tool. The development and test process in tracked in Jira tool (Attachment 2). Attachment 2 available at the end of this thesis shows how Jira tool is used as Kanban board in managing development and testing tasks.

The Figure 5 shows how development and test process goes in BroadSoft. Typically in every release there will around 5 to 6 new features will be released. For designing new features a feature specification document is taken as reference. Both development and test process starts taking feature specification document as reference in writing code and writing test-cases. In test process one of the QA person writes test-cases and these test-cases will be reviewed in a review-meeting and then test-cases will modified according to review comments. Test cases will be executed for a new feature and if any issues found in testing phase these issue will be tracked in Jira tool. Once the product gets stabilized during the process and meets specification then product will be released.

4.2 Automation Test Process

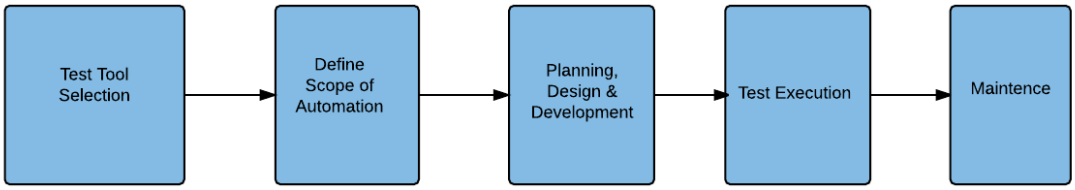


Figure 6. Automation Test Process, Adapted from Automated Testing (Guru99.com, 2016)

Figure 6 shows how test automation process goes in general from tool selection to maintenance of test automation. Even though Figure 6 didn't give initial phases of test automation process it gives details once test automation goes to mature state. Test automation process usually process starts with feasibility of application automation or not then goes tool selection (Tutorialspoint.com, 2016). Tool selection depends on which technology application is running. After selecting a tool then proper framework is needed, next chapter 5 I will give some of the popular available frameworks. After framework selection building proof of concept (POC) with end to end scenario to evaluate if tool can support automation of application. After building the POC, framework development is carried out, which is a crucial step for the success of any test automation project. Framework should be built after diligent analysis of the technology used by the application and also its key features (Tutorialspoint.com, 2016). The scripts are developed and executed, executed results are analyzed and defects are logged if any. Test scripts usually maintained in version control.

4.2.1 BroadSoft Automation test process

To know more about current BroadSoft test automation process it is important to know how test automation project start to current situation. Project was started with test automation tool selection, a Squish tool (Froglogic, 2016) is selected for test automation. We have started writing test cases on top of squish, we are able automate certain number of test cases. We don't have any framework at that moment so depend on tool feature functionality in writing test-cases and found that writing and maintenance of test cases became difficult. Writing test cases for non-programming background became bit difficult.

We have chosen Robot Framework (Robot Framework, 2016) on top of the squish tool (Figure 7). The current project has test automation framework and BroadSoft product is using Robot framework which is used for writing test-cases. At present a non-programming background also able to write test-cases. QA team is writing test-cases on top of the Robot

Framework using keyword driven style and test cases are maintained in Git version control (Git, 2016).

The current project is not that matured of maintenance of test-cases, project is initial phase of developing new automation test cases for existing manual test-cases. This project needs some improvement in handling of new feature test automation and maintenance of test-cases, maintenance involves maintaining of test scripts, running test scripts, analyzing and creating Jira tickets if there are any issues in the product. Test automation project is still new so there is no such test automation process in BroadSoft, but there is test automation framework as show in Figure 7 is used for test automation project.

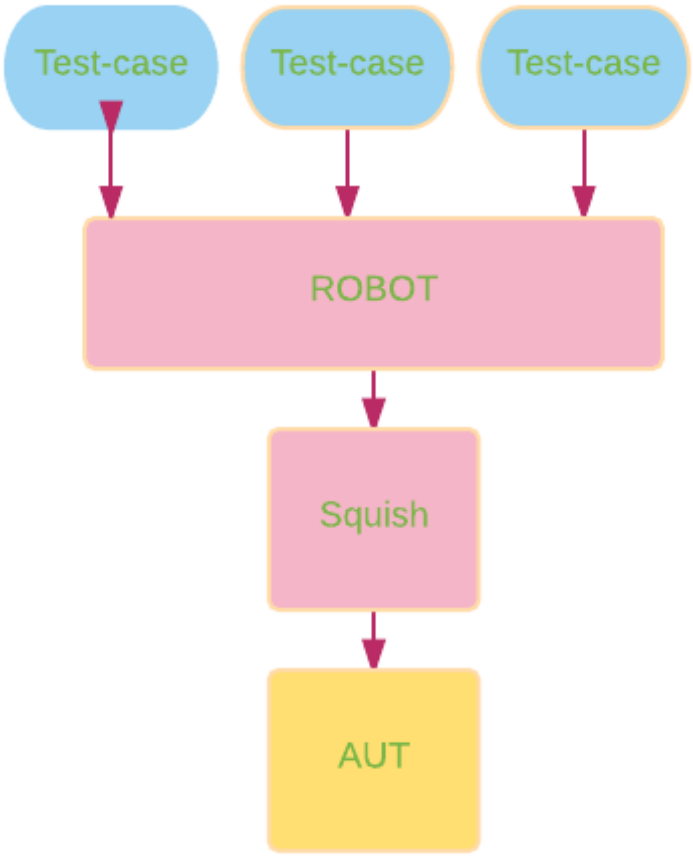


Figure 7. Automation testing framework at BroadSoft.

We wrote around 100 test cases using this current new framework. These test-cases were not properly integrated to current QA process. In regression test-cycles run we totally depend on manual test-case even though we have automation test-cases to certain extend. This research focus how we full utilize test automation in regression test-cycles also. We are writing test-cases to already released features. We don't have any process how we approach new feature test-cases, this will be covered in my research result available in section-5.2.

4.3 Robot Framework

As discussed in above sections we have chosen Robot Framework on top of the squish to write test-cases QA team writes test-cases on top of Robot Framework. Robot framework running on top of the product provides some advantages in writing, maintaining, running test-cases is more efficient than just using squish tool on top of the product. Let's discuss some of the advantages of Robot framework. Robot Framework is a Python-based, extensible keyword-driven test automation framework for end-to-end acceptance testing and acceptance-test-driven development (ATDD). It can be used for testing distributed, heterogeneous applications, where verification requires touching several technologies and interfaces (Robot Framework User Guide, 2016). Let's discuss some theoretical part of Robot framework.

4.3.1 Keyword-driven/Table-driven testing

Keyword-driven testing or some call it table-driven testing are the notions widely applied to an application-independent automation. The tester needs to develop data tables with keywords, independent of the test automation framework or any other tool used to run them. Then it is required to code the test script that will, in its turn "drive" the tested application and the data. Tables in a keyword-driven test will contain the information on the functionality of the tested application and step-by-step instructions for each test (Robot Framework - Test Automation the Smart Way, 2016).

The keywords can be derived as 'High level keywords' for define application level test-cases and 'Low level keywords' to keep test-cases minimal some keywords created called low level keywords which is used from high level keywords and then 'Technical keywords' these are keywords which actual accessing system under test.

In BroadSoft environment we have 'High level keywords' for writing actual test-cases which will look like manual test-cases and 'Low level keywords' which are actual keyword implementation and calling squish level scripts. 'Technical keywords' are written in python which will call the Application under test using Squish tool.

4.4 Robot Test Cases

Robot test-cases as many advantages in terms of writing, managing, running test-cases, let see these in details.

4.4.1 Writing Test Cases

Robot test-cases are written using keyword style. These test-cases are stored in TSV (table separated value) files or in HTML.

In BroadSoft we are writing in keyword style for test cases and these keywords stored in robot file format.

4.4.2 Managing Test Cases

Robot framework allows different ways to organize test-cases. Test cases can be made in to different test suits, test suits contains different test cases stored in files with robot, html or TSV extension. Robot framework allows the test case be marked with ‘tags’. Each test case can be tagged as per its business scope or importance of test case, example tags are critical, regression, quick etc. During executing tests can be executed based on tag style.

In BroadSoft we are managing test cases by storing all related test cases in to one test suite and all these test cases are marked with tags. Some tags we are using as critical, regression, feature specific tags. Test cases are executed using only tag style.

4.4.3 Running Test Cases

To run written test case, robot framework allows to execute single test case, allows to execute whole test case file or test suite and it allows to execute in tags style to run test cases. The execution results are generated in different formats.

In BroadSoft all test cases are executed using only tag style. After execution, tool will generates different files (log, output, report) with html extension. These files will give more details about passed and failed test cases. Attachment 4 and 5 at the end of this thesis will show execution results of BroadSoft which are took from Jenkins job.

4.5 Current BroadSoft regression test process

To improve BroadSoft regression test process it is important to study existing regression test process and how regression testing is used in different releases. At BroadSoft every year release different version of product, some releases includes new feature release and some releases are maintenance releases. Typically in year around 7 releases which includes feature and maintenance releases. For every release QA team executes regression test-cases. These regression test-cases are manual test-cases. There are more than 1500 test cases. Each release contains sub releases like VEA (very early adopters), EA

(early adopters), and RC (Release Candidates). There are typically around 5-7 sub-releases for each main release. Below Figure 8 show how sub-releases are done before main-release.

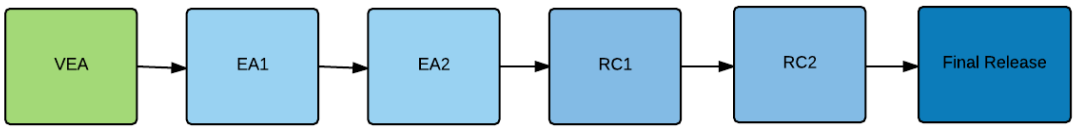


Figure 8: Release process at BroadSoft.

4.5.1 Regression Testing

In year 2015 we have released 7 releases, so for each release QA team runs regression test-cases so total of 38 regression test-cycles during year. In the year we have planned 5700 test-cases and out which we have executed 3300 test-cases, all these test-cases were executed using manual method. Typically regression testing time period will take around 3-4 weeks.

4.5.2 Regression testing test data

I have collected test data from test-link tool for the year 2015, this data is used for basis for regression testing improvement.

Releases	7
Total no of cycles	38
Planned Test cases	5734
Executed Test cases	3229
Time period	1 year
No-of persons	2-4

Table 1: BroadSoft Regression testing in the year 2015.

The above Table 1 show number of total number releases and total number of regression plans executed in a one year. This data is used for analysis and more details is covered in chapter 6 regression testing and improvement ideas.

5 Test Automation Frameworks and improvement ideas

To recommend test automation framework which can be fitted current test process there is need to study different popular test automation frameworks available. I have studied different frameworks and chosen proper framework which fits in to current process, different frameworks has their own advantages and disadvantages. This section will provide different frameworks available in current market and the frameworks chosen which will fit into current process. This section also give improvement ideas for test automation process using this chosen test automation framework.

Before discuss different test automation framework it is necessary to know what is framework and what is test automation framework. A 'Framework' is considered to be a combination of set protocols, rules, standards and guidelines that can be incorporated or followed as a whole so as to leverage the benefits of the scaffolding provided by the Framework (Softwaretestinghelp, 2016).

A 'Test Automation Framework' is scaffolding that is laid to provide an execution environment for the automation test scripts. The framework provides the user with various benefits that helps them to develop, execute and report the automation test scripts efficiently. It is more like a system that has created specifically to automate our tests (Softwaretestinghelp, 2016).

5.1 Types of Test Automation Framework

There are many different varieties of frameworks available in current market. These different frameworks has their own advantages and disadvantages. We will focus framework which will easy in maintenance and reusability and mostly fit in current BroadSoft test process.

Some of test automation frameworks available at present situation are (Softwaretestinghelp, 2016):

1. Module Based Testing Framework
2. Library Architecture Testing Framework
3. Data Driven Testing Framework
4. Keyword Driven Testing Framework
5. Hybrid Testing Framework
6. Behavior Driven Development Framework

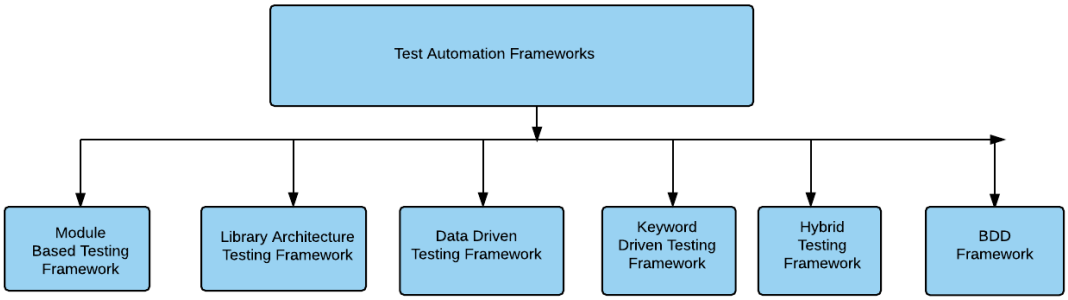


Figure 9. Different Test Automation Frameworks, Adapted from (Softwaretestinghelp, 2016)

5.1.1 Module Based Testing Framework

Module based Testing Framework is based on one of the popularly known OOPs (Object Oriented Programming) concept of Abstraction. The framework divides the entire “Application under Test” into number of logical and isolated modules. For each module, we create a separate and independent test script. Thus, when these test scripts taken together builds a larger test script representing more than one modules (Softwaretestinghelp, 2016).

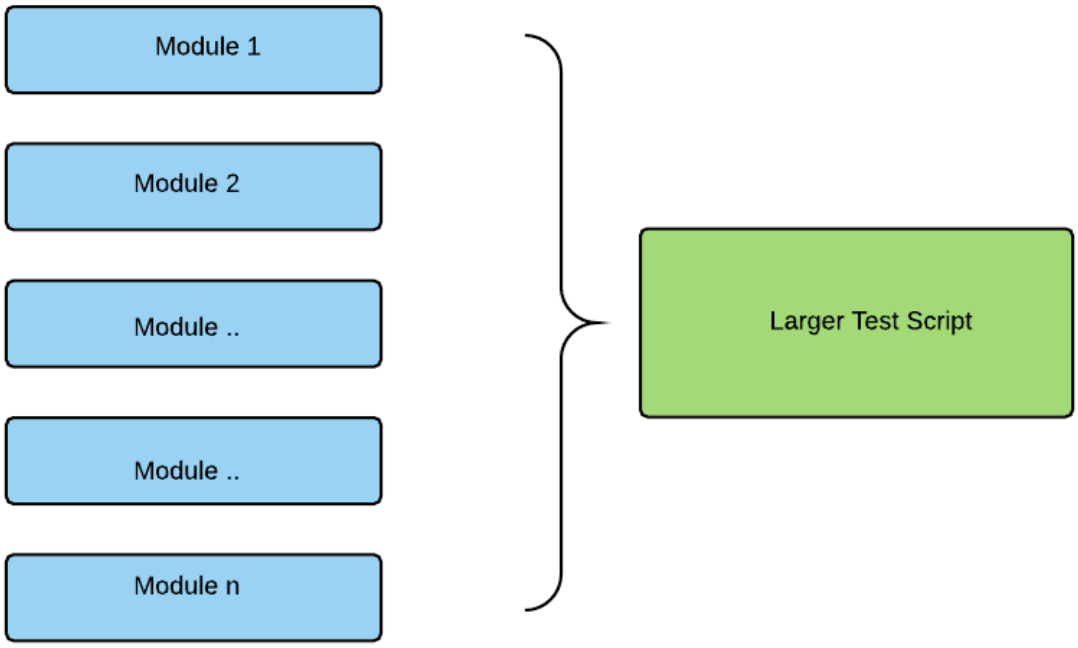


Figure 10. Module Based Test Automation Frameworks, Adapted from (Softwaretestinghelp, 2016)

Test Automation project view:

This framework is easier and cost effective in maintenance. This framework embeds test data into test scripts, thus whenever we want to use different test data it requires to manipulate all test scripts. This framework is not chosen for test automation project because of bit older test automation framework and needs programming skills for writing and maintenance of test-cases.

5.1.2 Library Architecture Testing Framework

The Library Architecture Testing Framework is fundamentally and foundationally built on Module Based Testing Framework with some additional advantages. Instead of dividing the application under test into test scripts, we segregate the application into functions or rather common functions can be used by the other parts of the application as well. Thus we create a common library constituting of common functions for the application under test. Therefore, these libraries can be called within the test scripts whenever required (Softwaretestinghelp, 2016).

In this framework all common steps and functionalities are grouped in to a library and call this library from test script, Figure11 show how test scripts calling library functions.

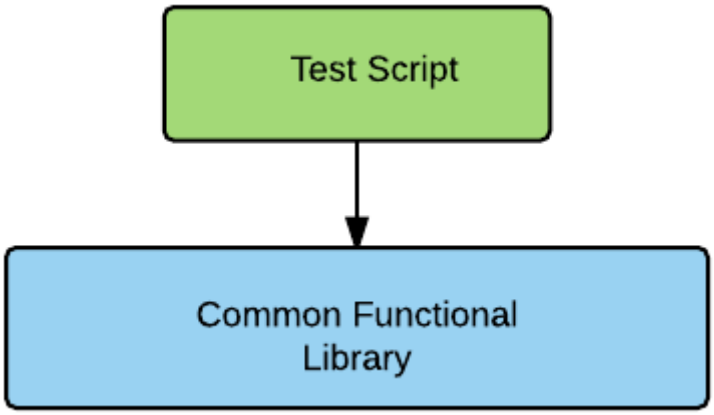


Figure 11. Library Architecture Test Automation Frameworks

Test Automation project view:

This framework has same advantages & disadvantages of module based test automation framework. This framework becomes little complicated because of introducing libraries. This framework is also not chosen current test automation project because this framework

needs some programming skills for maintain library and test scripts, which makes it bit difficult for QA persons to contribute to test automation process.

5.1.3 Data Driven Testing Framework

Data-driven testing is creation of test scripts where test data and/or output values are read from data files instead of using the same hard-coded values each time the test runs. This way, testers can test how the application handles various inputs effectively (Data Driven Testing 2016).

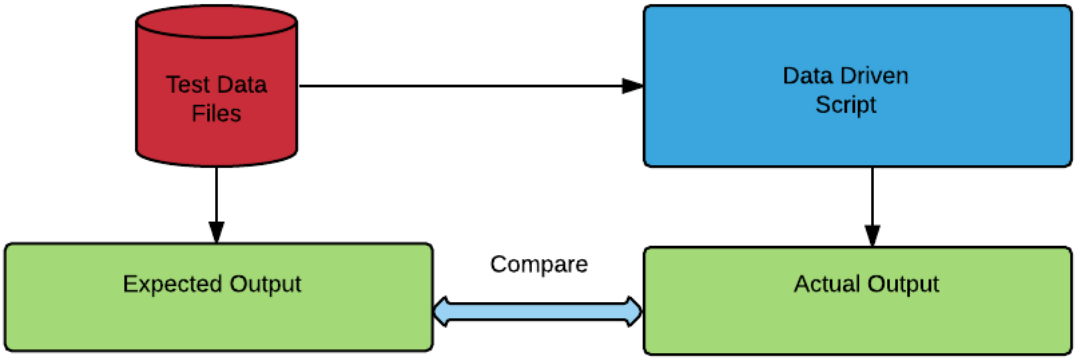


Figure 12. Data Driven Testing Framework, Adapted from (Tutorialspoint, 2016).

Test Automation project view:

Data driven framework is partly chosen for test automation project. Squish tool uses data driven approach. Figure 13 show how data driven testing framework used BroadSoft using squish tool. Test-Data provided to test-scripts using .tsv (Table Separated Value) files. Test scripts are written using squish tool. Squish tool has a capability of generating test-scripts by recording AUT. Generated scripts are not fully compatible with AUT, so lot of changes are made to scripts to be compatible with AUT. Writing test-cases and maintain test cases using data driven approach became bit difficult, so went to another framework called keyword driven testing framework that will be discussed in next section.

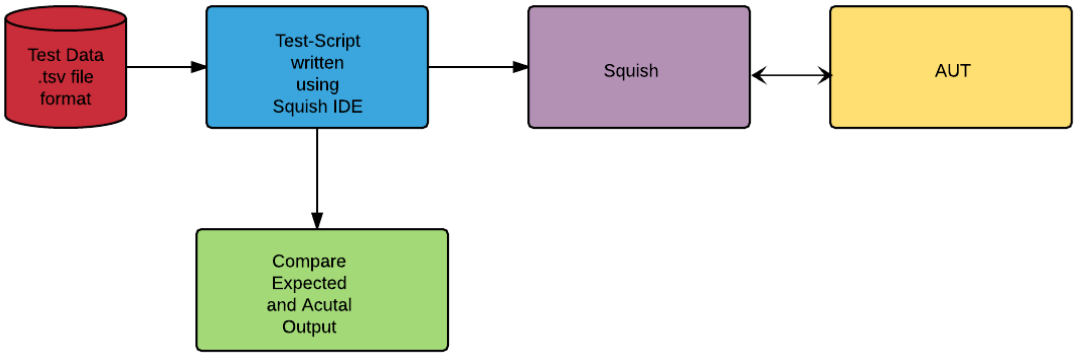


Figure 13. Data Driven Testing using squish at BroadSoft, 'AUT' is application under test which BroadSoft application.

5.1.4 Keyword Driven Testing Framework

The Keyword driven testing framework is an extension to Data driven Testing Framework in a sense that it not only segregates the test data from the scripts, it also keeps the certain set of code belonging to the test script into an external data file. These set of code are known as Keywords and hence the framework is so named. Key words are self-guiding as to what actions needs to be performed on the application (Most Popular Test Automation Frameworks, 2016).

The keywords and the test data are stored in a tabular like structure and thus it is also popularly regarded as Table driven Framework. Take a notice that keywords and test data are entities independent of the automation tool being used (Most Popular Test Automation Frameworks, 2016).

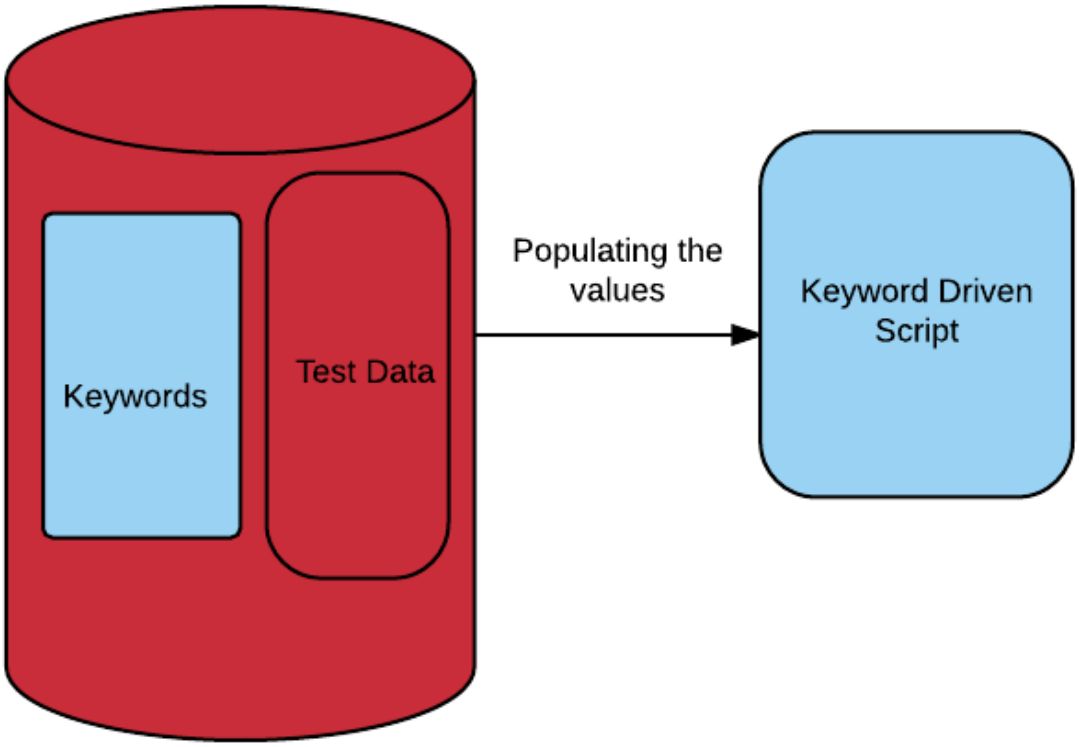


Figure 14. Keyword Driven Testing Framework, Adapted from (Softwaretestinghelp, 2016).

Test Automation project view:

Keyword driven style is used in current test automation project. Robot Framework will provides keyword driven testing approach. In this approach test scripts are written using keywords, these keywords calls in-build Robot libraries. Robot framework has different in-build libraries which can be used for creating test-scripts and running AUT. In Figure 15 shows how keyword style used in writing test scripts and running AUT with the help Robot framework.

In current test automation project case if we use only Robot framework on top AUT, it doesn't provide full functionality of running AUT. So another framework called hybrid test automation framework chosen which will be discussed in below section.

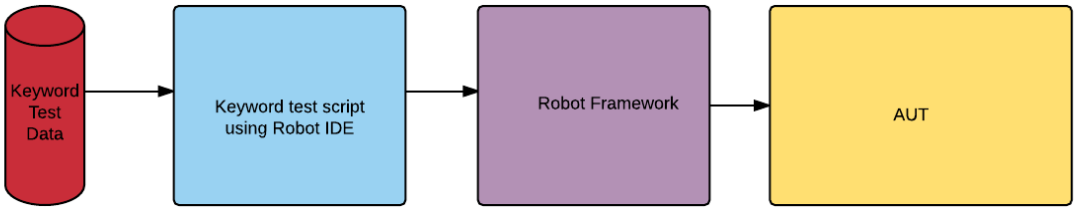


Figure 15. Keyword Driven Testing using Robot framework at BroadSoft Finland.

5.1.5 Hybrid Testing Framework

As the name suggests, the Hybrid Testing Framework is a combination of more than one above mentioned frameworks. The best thing about such a setup is that it leverages the benefits of all kinds of associated frameworks.

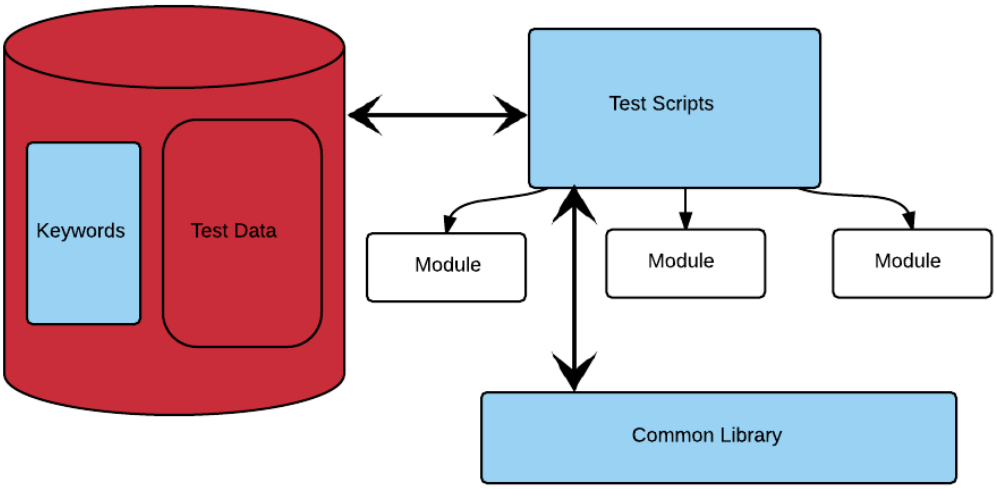


Figure 16. Hybrid testing framework, Adapted from (Softwaretestinghelp, 2016).

Test Automation project view:

Hybrid test automation is chosen to current test automation project. Hybrid test automation is combination of data driven test automation and keyword driven frameworks. In project case Squish and Robot tools used in achieving this framework. Figure 17 shows hybrid test automation framework which is combination of squish and robot tool, test scripts are written using robot framework and some property modules written to call squish which internally calls AUT.

This framework makes easy for non-programming background to write scripts, now QA team writing test scripts using Robot framework style and modules are written by DevOps (Development and operations) team.

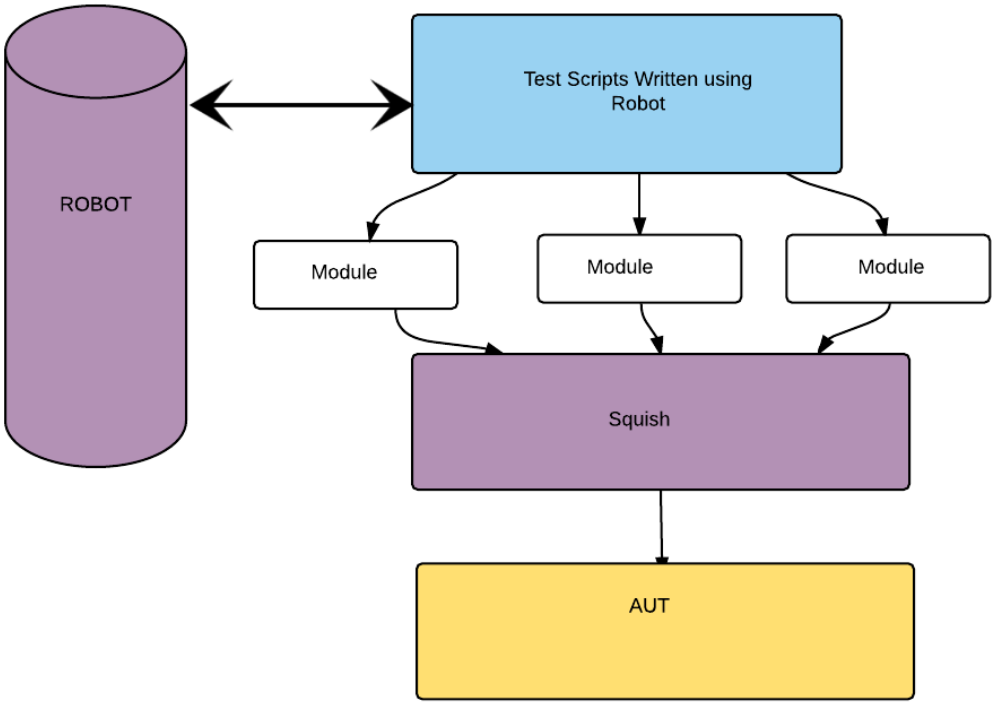


Figure 17: Hybrid testing framework at BroadSoft.

5.1.6 Behaviour Driven Development Framework

Behavior Driven Development framework allows automation of functional validations in easily readable and understandable format to Business Analysts, Developers, Testers, etc. Such frameworks do not necessarily require the user to be acquainted with programming language. There are different tools available for BDD like cucumber, Jbehave etc. (Softwaretestinghelp, 2016).

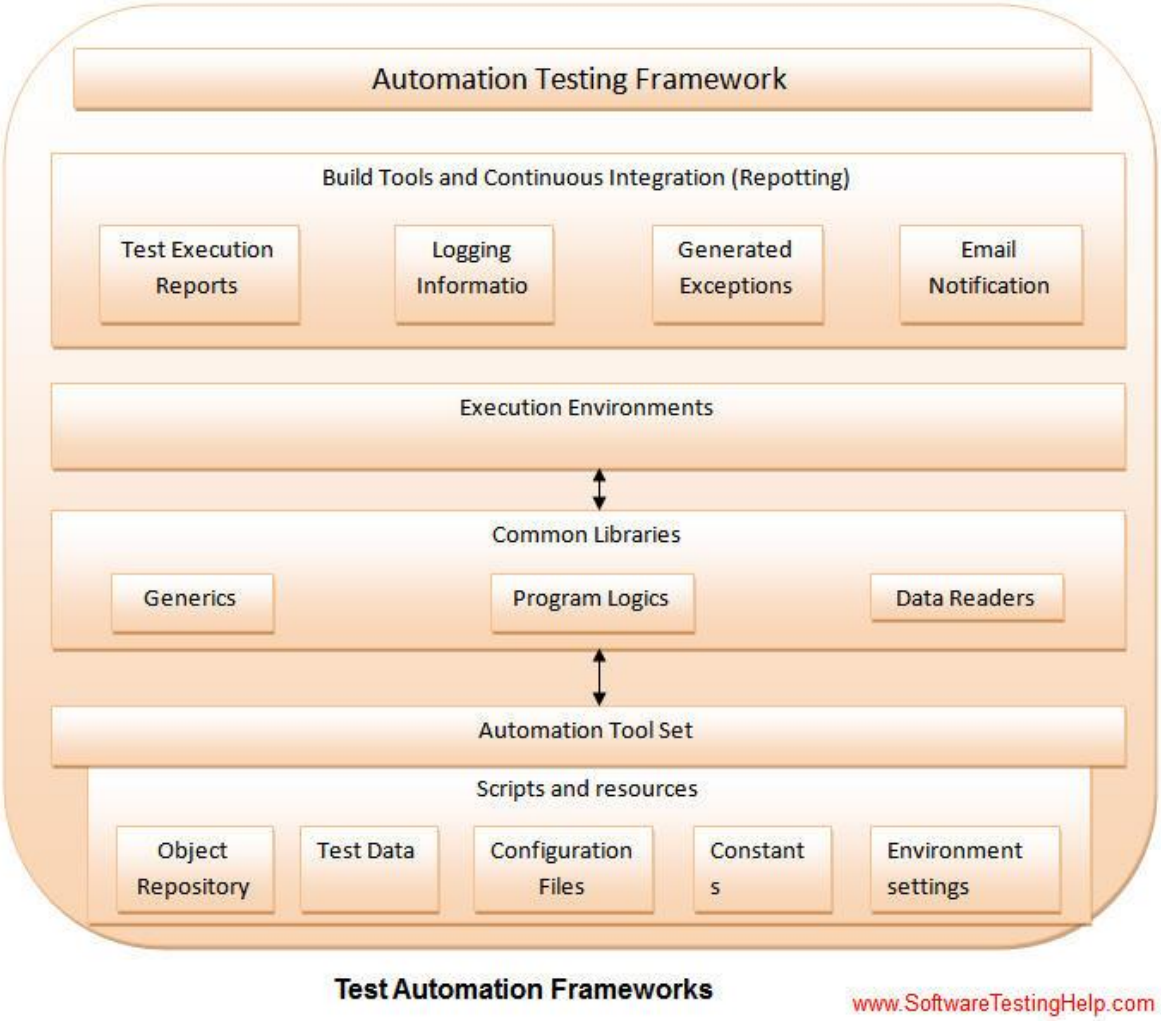


Figure 18. Behavior Driven testing framework took from SoftwareTestingHelp.com.

Though the above pictorial representation of a framework is self-explanatory but we would still highlight a few points (Softwaretestinghelp, 2016).

1. **Object Repository**: Object Repository acronym as OR is constituted of the set of locators types associated with web elements.
2. **Test Data:** The input data with which the scenario would be tested and it can be the expected values with which the actual results would be compared.
3. **Configuration File/Constants/ Environment Settings**: The file stores the information regarding the application URL, browser specific information etc. It is generally the information that remains static throughout the framework.
4. **Generics/ Program logics/ Readers**: These are the classes that store the functions which can be commonly used across the entire framework.
5. **Build tools and Continuous Integration**: These are the tools that aids to the frameworks capabilities to generate test reports, email notifications and logging information.

Test Automation project view:

Behavior is not fully implemented in current test automation framework. Figure 18 shows different levels of behavior driven testing frameworks, in this framework test-cases are behavior driven style using keywords like Given, When, Then (Codecentric, 2009). The 'Given' is used as pre-condition, When is used as behavior under test and Then is expected result of behavior. The current test automation project test cases using keyword driven style and data driven style, but this improved to have behavior driven style. The behavior driven style most convent to read for business analyst, developer and tester. BDD style used in some extend in current project, there are some missing features such as email notification if there any test cases failure it should be notified to particular persons which helps in maintenance of test cases.

5.2 Key Driven Test Process and Improvement ideas.

Why do test automation project fails?

Automation testing is an extra addition to current test process. According to Cem Kaner (Cem Kaner), "Improving the Maintainability of Automated Test Suites", it can take between 3 to 10 times longer to develop, verify, and document an automated test case than to create and execute a manual test case. This is especially true if you select to use the "record/playback" feature. So there should be always sanity check is needed for every test automation to successful. In current test automation is also there needs to be measurement need for how much time spend on creating automation test-case by expert, functional testers and management over heads etc. If number of hours spend is more than manual test process then test automation project is consider as failure project, since this project is initial phase some improvements can be done in:

5.2.1 Process

Test automation requires changes in current test process. Changes includes 'Test design approach', test automation requires more specific design than manual test design. 'Test coverage', test automation requires more specific functions to be tested in different scenarios for test coverage. 'Test execution', functions which are tested using automation should not be tested manually. Current test automation project needs to apply these changes for effective use of test automation.

5.2.2 Maintenance

Current test automation requires maintenance of test-cases, maintenance includes updating test scripts when functional changes in AUT and technical maintenance needed to existing test scripts and for test environment set-up.

5.2.3 Expertise

In order to efficient automate tests, test automation experts are required, good test engineers, system experts and great SW developers.

5.2.4 Keywords

The keywords are the basic functions to test AUT. There are many types of keywords some keywords do specific action and some keywords do utility function such wait certain period of seconds etc. In most keywords requires parameters, keywords much be designed considering default parameters. In this test automation project keywords needs to be reviewed and updated accordingly.

Some of KDT (Key Driven Testing) principles like a) simplicity of test cases implementation and reviewing, b) test automation infrastructure should be separated from logical layer so that tester automate test-cases before application is ready. In order to maximize the benefit from KDT, a plan to be used to assess the on-going effectiveness of the automation program (Ayal Zylberman, 2016)

In Current test automation project we are focus on automating existing regression test cases this is usual tendency for any test automation project. Usually there is tendency that test automation cannot be created until application is ready then it leads to automate only regression test cases. This approach also leads to parallel execution of manual and automation test cases. KDT allows testers to plan test automation before application is ready. I recommend current test automation project should use KDT approach to automate new feature rather than focus on only regression test-cases. To focus on KDT approach for new feature, an organization structure and responsibility is needed like who will define the keywords and who will develop keywords and who will develop and execute keywords. Figure 19 gives an example how KDT approach can be used for new feature, this approach can be used in current test automation project. In this approach tester defines keywords based on feature spec. If keywords are missing then keywords needs to develop by test automation team, in this project we need to create Jira ticket for creating new keywords. Once the keywords are ready whole test-scripts is written for that feature and executed in a test environment and test-scripts are transferred to regression repository.

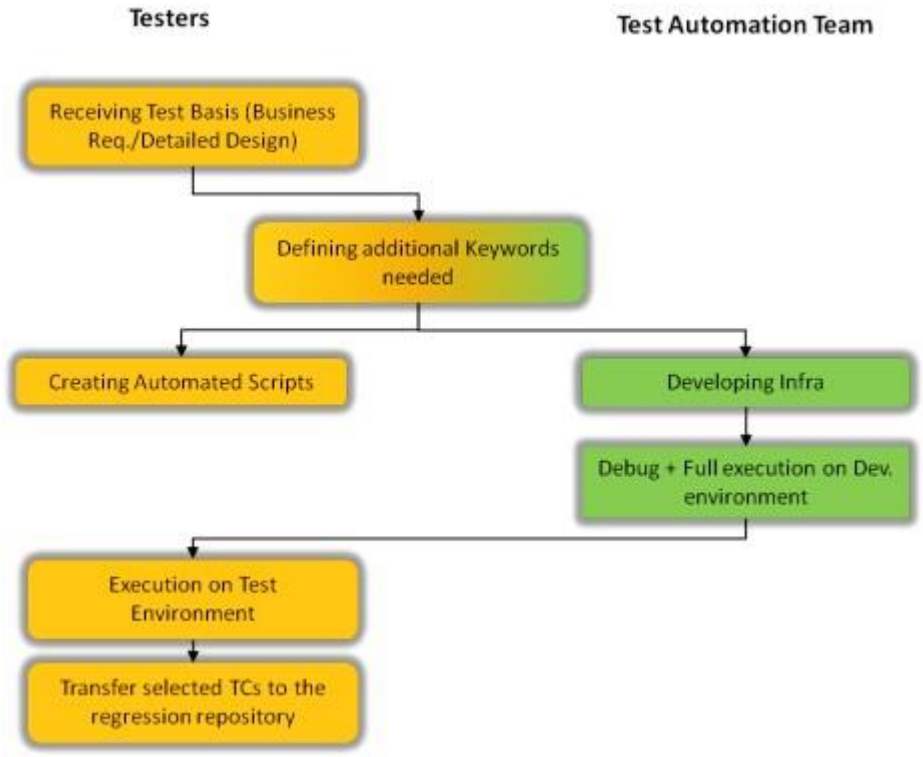


Figure 19. Keyword driven test (KDT) automation process (Ayal Zylberman, 2016)

6 Regression testing and improvement ideas

This section is about regression testing and improving regression testing at BroadSoft. To improve regression testing at BroadSoft, I have studied some of the tools and methods to improve testing which can be applied in improving regression testing at BroadSoft.

When any modification or changes are done to the application or even when any small change is done to the code then it can bring unexpected issues. Along with the new changes it becomes very important to test whether the existing functionality is intact or not. This can be achieved by doing the regression testing (What Is Regression Testing in Software).Regression testing sometimes a headache but it is important part of overall test effort on each application release. They are required because they detect problems that have been undetected upstream. However they also often miss bugs which are found after deployment when the cost to fix them is usually estimated to be at least 10 times higher (Testing Effectiveness by 30%). Below Figure 20 show in a time-line how regression test-cases increase with each release.

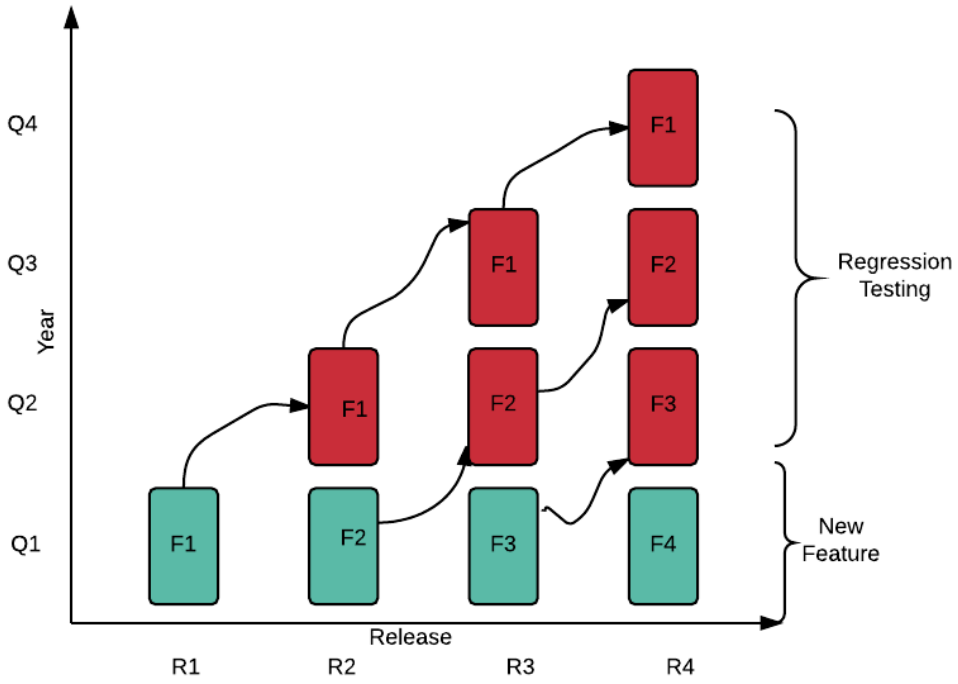


Figure 20. Regression testing in a timeline (Releases vs year), Adapted from coverity blog.

In BroadSoft Finland case in the year 2015 we have increased from 1000 test cases to around 1500 test-cases, total of 500 test-case increased in the year 2015. Typically to run full regression test-cases takes 3-4 weeks. If we consider on average of 500 test-cases

increase then after 5 years there will be total of 4000 test-cases and it will take around 8-9 weeks to run full regression test-case. To avoid this kind of overhead of running regression test-cases we are automating test-cases but all test-cases cannot be automated by any tool, so there is always need of manual testing. The regression testing needs to be handled effectively. Regression testing effectiveness is required to (Testing Effectiveness by 30%):

1. **Reduce risk of bugs in each release:** By selecting right test to be execute in a proper dedicated time-frame for regression testing.
2. **Responsive to business needs:** Some releases needs to be fast and smaller release. This requires faster regression testing and so thus requires effective regression test-plans in the context of each version. In BroadSoft case there is need to different test-plans and these test plans needs to be executed based on release criteria.
3. **Adapt to financial constraints:** Regression testing effectiveness is required based on financial constraints.

6.1 Regression test strategy

There should need to have test strategy for any project for effectively use regression testing. I have studied different tools in that coverity tool (“Software Testing and Static Analysis Tool Coverity, 2016”) which can be used for BroadSoft regression testing purpose.

In general test strategy limits number of test by two types of tests:

1. Test only major features of the product regardless of requirements and risks. In BroadSoft case, test cases such as ‘smoke tests’ which will test only test major features of product.
2. Test features which are less robust i.e. where bugs found in previous regression releases.

With above two points it leads to run same test-cases each software release. Which makes difficult to detect regression bugs. The above two of test strategy limitation are applicable to BroadSoft regression test process. So there should some tool which can be used to improve regression test process, the tool is discussed in below section.

6.2 Kalistick Test Scoring tool and results

To improve this situation there should need of tools such Kalistick developed “Test Scoring” This tool is used when a new release of an application is done, whether corrections, changes, or new features, development teams are making changes to the application code. It is these changes that generate regressions on existing features (Software Testing and Static Analysis Tools, 2016).

6.2.1 Links between changes and tests

Coverity done research with collaboration from research center’s shows that the probability to find a bug with a specific test is directly related to the volume of code modification that this test will use when you execute it (Software Testing and Static Analysis Tools, 2016).

The impact analysis goal is therefore to establish links between changes and tests. This analysis is impossible without a tool, however, existing tools do some technical analysis but they don’t provide any result at the test level.

To meet this need, Kalistick have developed “Test Scoring”. At each release of the application, each test is scored according to the volume of changes that affects its execution. This score is based on three unique techniques:

1. Capturing Test Footprints. Whenever a tester executes a test, Kalistick’s system automatically captures all code executed within the application for this test. This footprint gives all the links between the test and the code; it is stored in a knowledge base.

2. Change detection. Kalistick scanner analyzes each release and detects changes by comparing the scan results with previous ones.

3. Kalistick engine analyzes Test Footprints to assess impact of changes and compute Test Scores.

The principle is the following: as soon as a test is executed the system stores it. At each future version it is scored to assess whether it is required to execute it.

Customers use the scoring within their usual testing tools like HP Quality Center, HP ALM, Test Link, XStudio, ReferTest, etc. Kalistick’s plug-ins provide seamless integration, so test teams can rely on it to boost the effectiveness of their test campaigns (Software Testing and Static Analysis Tools, 2016)

6.3 Results

It can increase test effectiveness over 30%. This is measured on two axes: gain in quality and time saving. Two customer's cases to illustrate these benefits (Software Testing and Static Analysis Tools, 2016):

1. A client executes its usual regression test suite and then it selects 10 tests to be executed using Kalistick scoring. Result: 3 regressions detected within 3 hours.

2. Another client measures the quality of each release according to the number of bugs detected in production within 3 months after deployment. Result on the first release tested using Kalistick, the number of bugs has been down by 50% leading to saving of tens of thousands dollars.

6.4 Regression improvements

Regression testing can be also improved by using existing tools such as testlink, Jenkins. Testlink supports keywords, existing manual test-cases should be marked with suitable keyword (E.g. 'Automated') once the test-case are automated, so that these test-cases can be ignored regression test plan.

6.4.1 Test Link plugin

At present situation project don't support integration of 'testlink' and 'Jenkins'. There is possibility to integrate testlink with Jenkins (Test Link Plugin, 2016) and DevOps need to integrate testlink in existing Jenkins job (Where Does QA Fit In DevOps, 2016). Testlink and Jenkins integration will make to view different test plans and number of test cases passed failed and blocked (Attachment 6). The testlink integration will useful for manual regression test cases and automation test cases.

7 Recommendation and development ideas

In this section I will conclude my recommendation and results of research in brief and improvement ideas of test automation process. The main aim of this thesis is test automation process improvement, the process itself needs continuous improvement to achieve effectiveness and efficiency of IT processes and services.

Test automation project itself is change from existing manual test process, so proper 'change management' is needed. Change management includes many sub-section like building the team for change, put the vision in right direction, communication of change etc. Change management is a continuous process, till now project is in right direction but it needs lot effective change management for moving from manual test process to automation test process. Test automation as change should not affect existing functionality or process of the whole product.

Test automation helps in regression testing to be done faster way, but the project need to make sure test automation should not effect in existing release process. The test automation should meet existing customer demands, it shouldn't break IT operations who support current product.

Test automation is a new project so knowledge should be documented and stored in proper place. The documentation should be easily accessible and communicated within the team to aware of such documents. There should proper knowledge management is need so that new person shouldn't necessarily rediscover knowledge.

The current test automation framework is chosen as a hybrid test automation framework which is combination of keyword driven framework and data driven frameworks. I recommend that the keywords written should be effectively written even if there is change in software language used for AUT, it shouldn't affect too much in rewriting existing keyword driven style test cases.

The BDD style is not used in current test automation project but BDD style can be incorporated in to current test cases using existing Robot framework tool, this will give add on advantage to different customers or business analysts to read our automation test case and understand. I recommend project should incorporate BDD style in writing test cases.

The current test automation process approach is writing test scripts for existing regression test cases, there should be plan to automate even to new features. The KDT approach which discussed in this thesis can be applied to automate new features. I recommend KDT approach should be used for upcoming new features.

There are some areas of improvement in the area of regression testing such as, when test cases are automated those cases should be marked in testlink tool. Even when writing manual test cases, manual test cases should be marked, which can be automated and which will go to regression test plan. Kallistick test scoring tool which discussed as part of this thesis can be used for improving regression testing. Test automation project don't have integration of 'testlink' with 'Jenkins', if this is integration it helps in improving regression testing and also used for effective test management.

In future if some want to extend research they can research on how much level of manual work reduced due to automation testing, this can be done only after 1year by that time automation test project will mature to compare both manual and automation testing. There are some other areas where research can be done like how to setup remote test environment and how effectively manage test automation etc.

References

Advantages of Automation, 2016 “Advantages of Automation.” Accessed April 5, 2016. http://www.softwaretestingmentor.com/automation/advantages-of-automation/.

Istqbexamcertification.com, 2016 “What Is Software Testing?” Accessed May 5, 2016. http://istqbexamcertification.com/what-is-a-software-testing.

Guru99.com, 2016 “Automated Testing: Process, Planning, Tool Selection.” Accessed May 5, 2016. http://www.guru99.com/automation-testing.html.

Tutorialspoint.com, 2016. “QTP Automated Testing Process.” Www.tutorialspoint.com. Accessed Feb 15, 2016. http://www.tutorialspoint.com/qtp/qtp_test_automation_process.htm.

Froglogic, 2016. “Froglogic • Squish GUI Tester • Squish for Qt.” Accessed May 5, 2016. https://www.froglogic.com/squish/gui-testing/editions/qt.php.

Robot, 2016. “Robot Framework.” Accessed May 5, 2016. http://robotframework.org/.

Git, 2016. “Git - About Version Control.” Accessed May 5, 2016. https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control.

Robot Framework - Test Automation the Smart Way, 2016. “Robot Framework - Test Automation the Smart Way!” Accessed March 28, 2016. http://quintagroup.com/cms/python/robot-framework.

Robot Framework User Guide, 2016. “Robot Framework User Guide.” Accessed March 28, 2016. http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html.

Softwaretestinghelp, 2016, Softwaretestinghelp “Most Popular Test Automation Frameworks With Pros And Cons Of Each – Selenium Tutorial #20.” Accessed March 28, 2016. http://www.softwaretestinghelp.com/test-automation-frameworks-selenium-tutorial-20

Cem Kaner. “Software Test Automation A Real-World Problem”, Accessed April 28, 2016: A Real-World Problem,http://kaner.com/pdfs/allegheny.pdf

Ayal Zylberman, 2016. Ayal Zylberman and Aviram Shotten “Practical Experience in Automated Software Testing.” Accessed April 13, 2016. http://www.methodsandtools.com/archive/archive.php?id=33.

Test Link Plugin, 2016. “TestLink Plugin - Jenkins - Jenkins Wiki.” Accessed May 16, 2016. https://wiki.jenkins-ci.org/display/JENKINS/TestLink+Plugin.

Codecentric, 2009. “Given/When/Then And Example Tables Using the Robot Framework.” Codecentric Blog, November 16, 2009. Accessed May 16, 2016. https://blog.codecentric.de/en/2009/11/givenwhenthen-and-example-tables-using-the-robot-framework/.

“ITIL v3, 2014. ITIL (Information Technology Infrastructure Library)/Introduction - Wikibooks, Open Books for an Open World.” Accessed May 12, 2016. https://en.wikibooks.org/wiki/ITIL_v3_(Information_Technology_Infrastructure_Library)/Introduction.

Kemmis and McTaggart’s, 1998. Stephen Kemmis and Robin McTaggart. PARTICIPATORY ACTION RESEARCH, n.d. http://www.corwin.com/sites/default/files/upm-binaries/21157_Chapter_10.pdf.

ITIL V3 Application Support, 2008. http://www.itservicemanagement-itil.com/wp-content/downloads/ITIL-V3-Application-Support.pdf

ITIL V3 vs ITIL 2011. “ITIL V3 vs ITIL 2011”.Accessed May 12, 2016. https://itil2014.wordpress.com/tag/service-operation/.

Meyer, 2000. Accessed May 12, 2016. http://www.actaneurologica.be/acta/download/2009-2/02-Vallenga%20et%20al.pdf

kemmis and mctaggart, D. Coghlan, 2008, and M. Keating. “Research for Action and Research in Action. Processual and Action Research in Dialogue?” Irish Journal of Management 29, no. 1 (2008): 1–18.

Kemmis and McTaggart 2000. Stephen Kemmis and Robin McTaggart. PARTICIPATORY ACTION RESEARCH, n.d. Accessed Mar 12, 2016. http://www.corwin.com/sites/default/files/upm-binaries/21157_Chapter_10.pdf.

McDermott, Aoife Mary, D. Coghlan, and M. Keating. “Research for Action and Research in Action. Processual and Action Research in Dialogue?” Irish Journal of Management 29, no. 1 (2008): 1–18.

Software Testing and Static Analysis Tools, 2016. “Software Testing and Static Analysis Tools | Coverity.” Accessed May 12, 2016. http://www.coverity.com/.

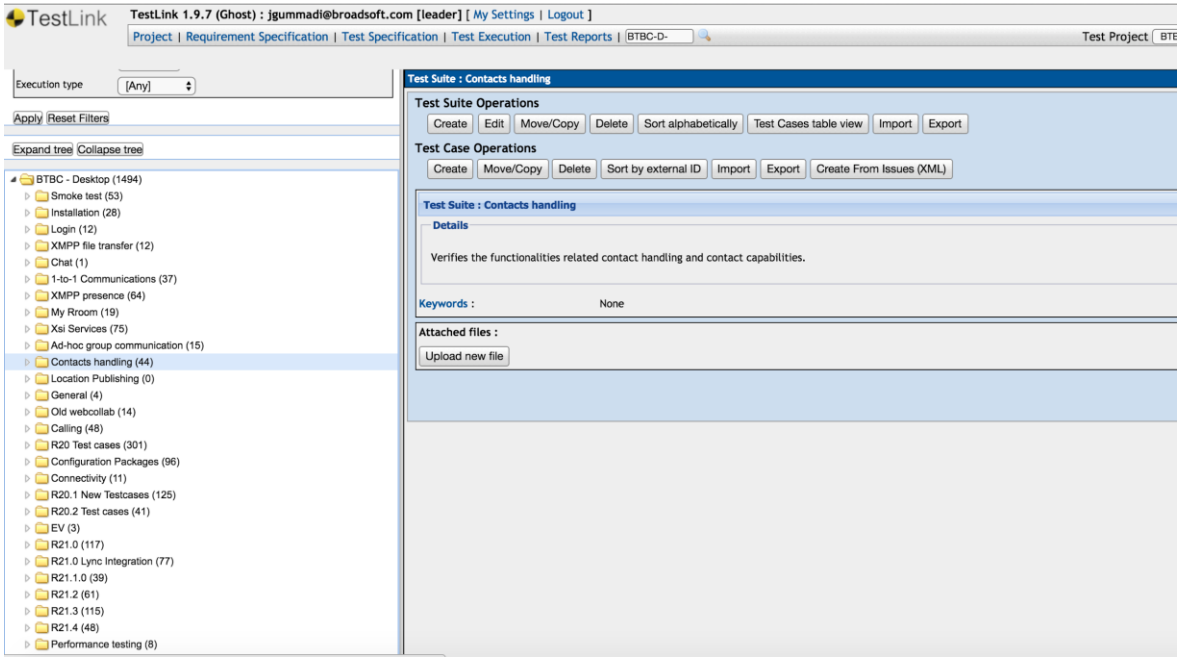
What Is Regression Testing in Software? 2016. “What Is Regression Testing in Software?” Accessed May 12, 2016. http://istqbexamcertification.com/what-is-regression-testing-in-software/.

Testing Effectiveness by 30, 2016. “How to Improve Regression Testing Effectiveness by 30%?” Accessed March 28, 2016. http://blog.coverity.com/2013/01/30/improving-regression-testing-effectiveness/#.Vvj74sl95p9.

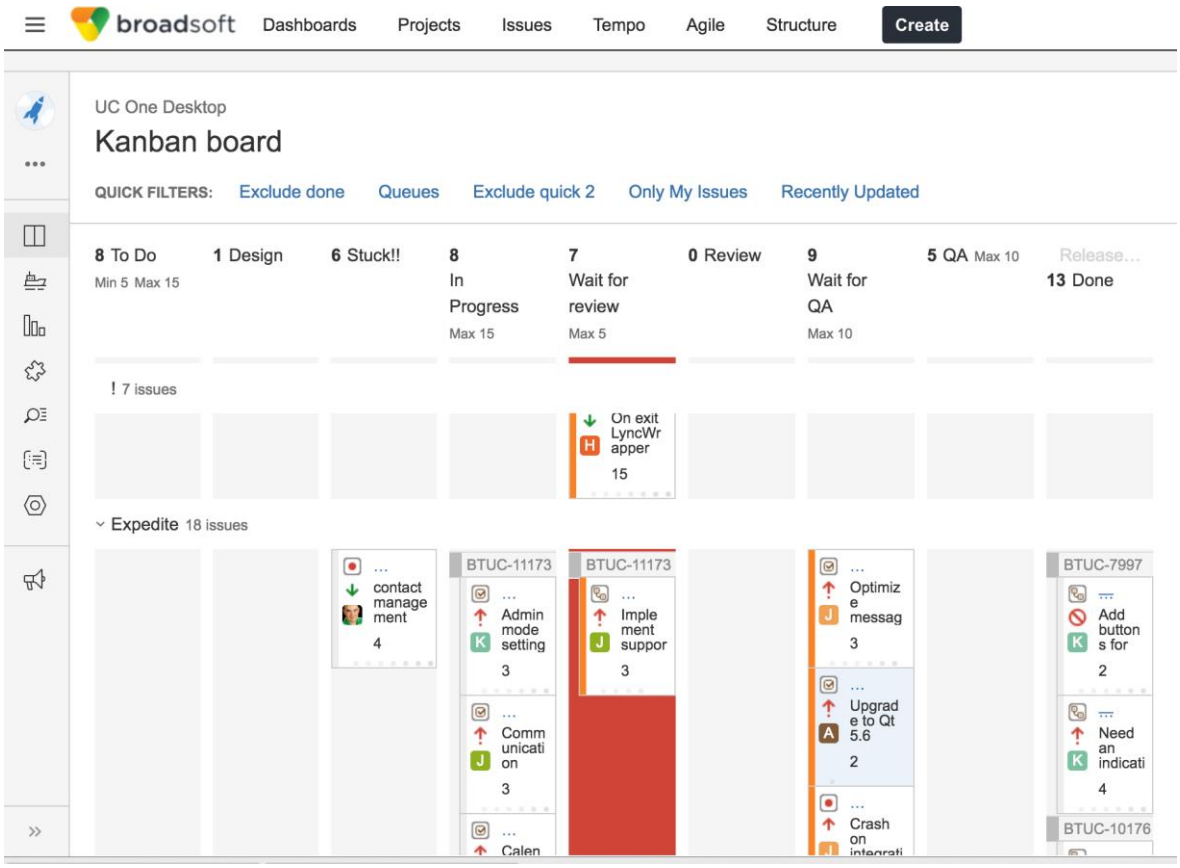
Where Does QA Fit In DevOps, 2016. “Where Does QA Fit In DevOps?” Accessed April 13, 2016. http://www.neotys.com/blog/where-does-qa-fit-in-devops/.

Attachments

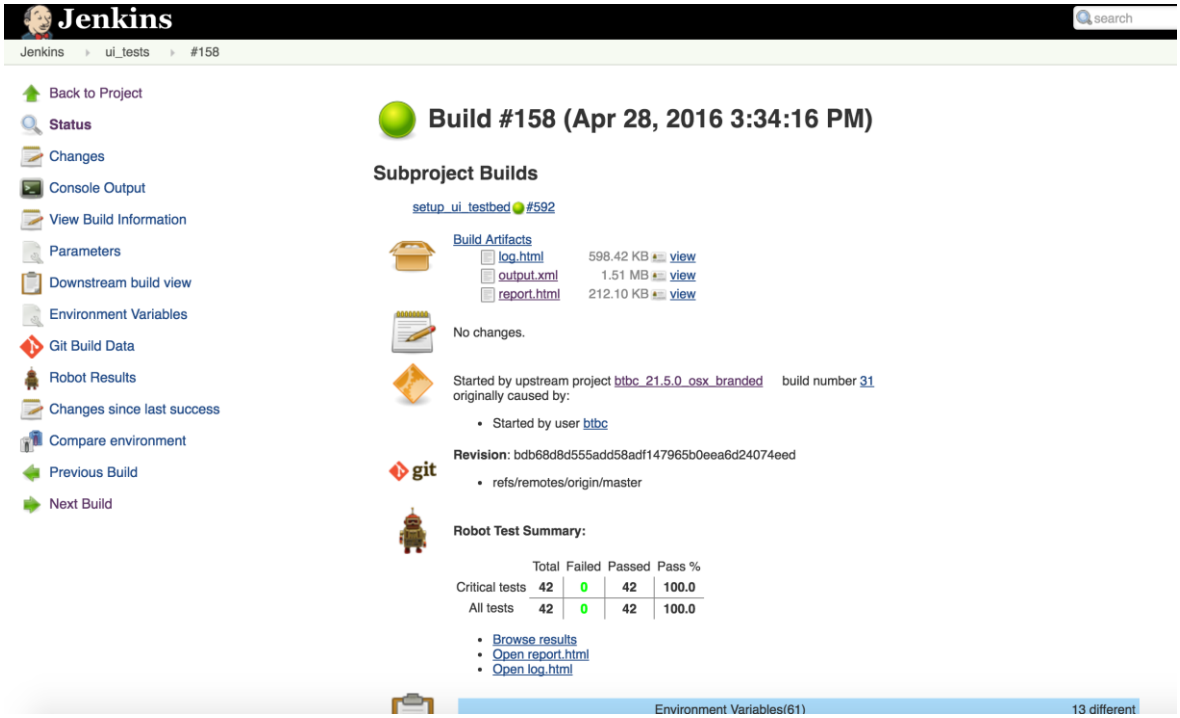
Attachment 1. Testlink for creating manual test-cases.



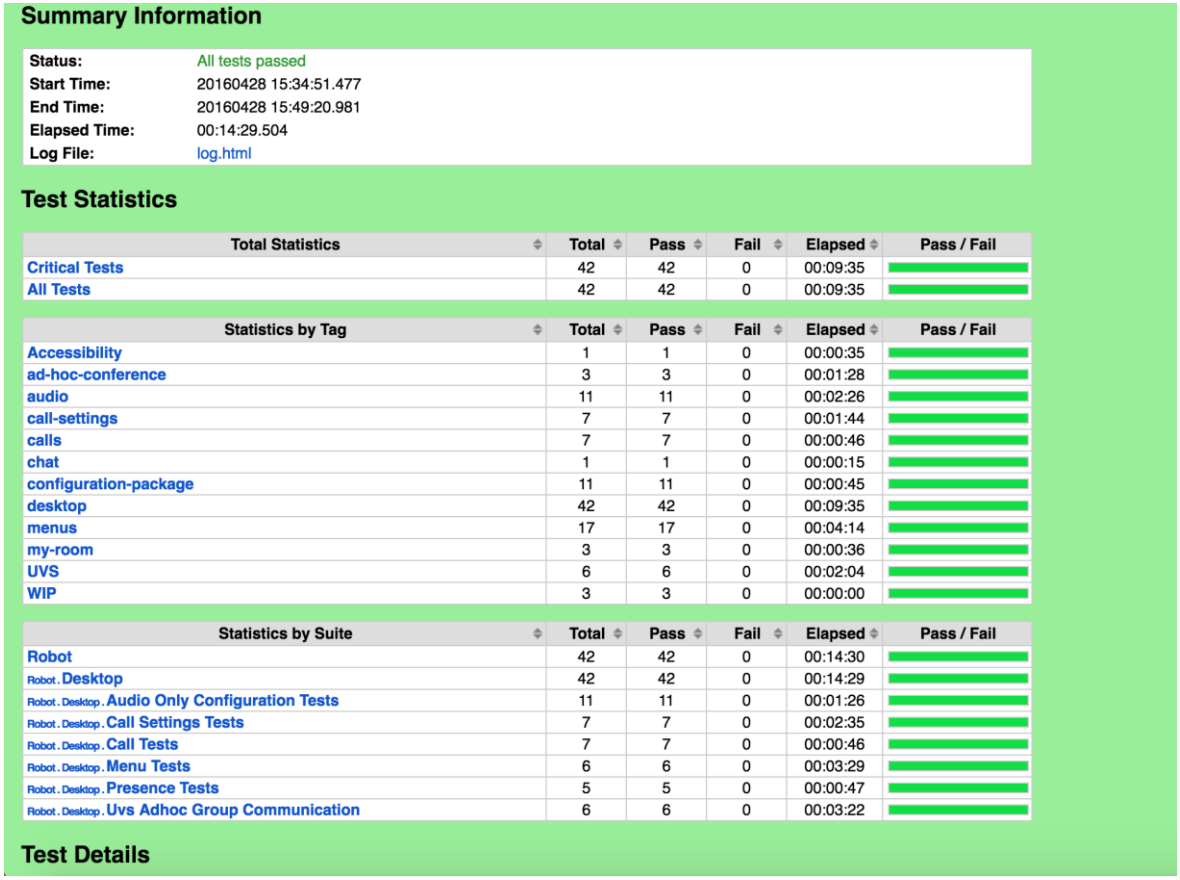
Attachment 2. Jira Tool used for tracking whole product process.



Attachment 3. Screenshot of Jenkins build job.



Attachment 4. Screenshot shows summary of test-cases from Jenkins.



Summary Information

Status:	All tests passed
Start Time:	20160428 15:34:51.477
End Time:	20160428 15:49:20.981
Elapsed Time:	00:14:29.504
Log File:	log.html

Test Statistics

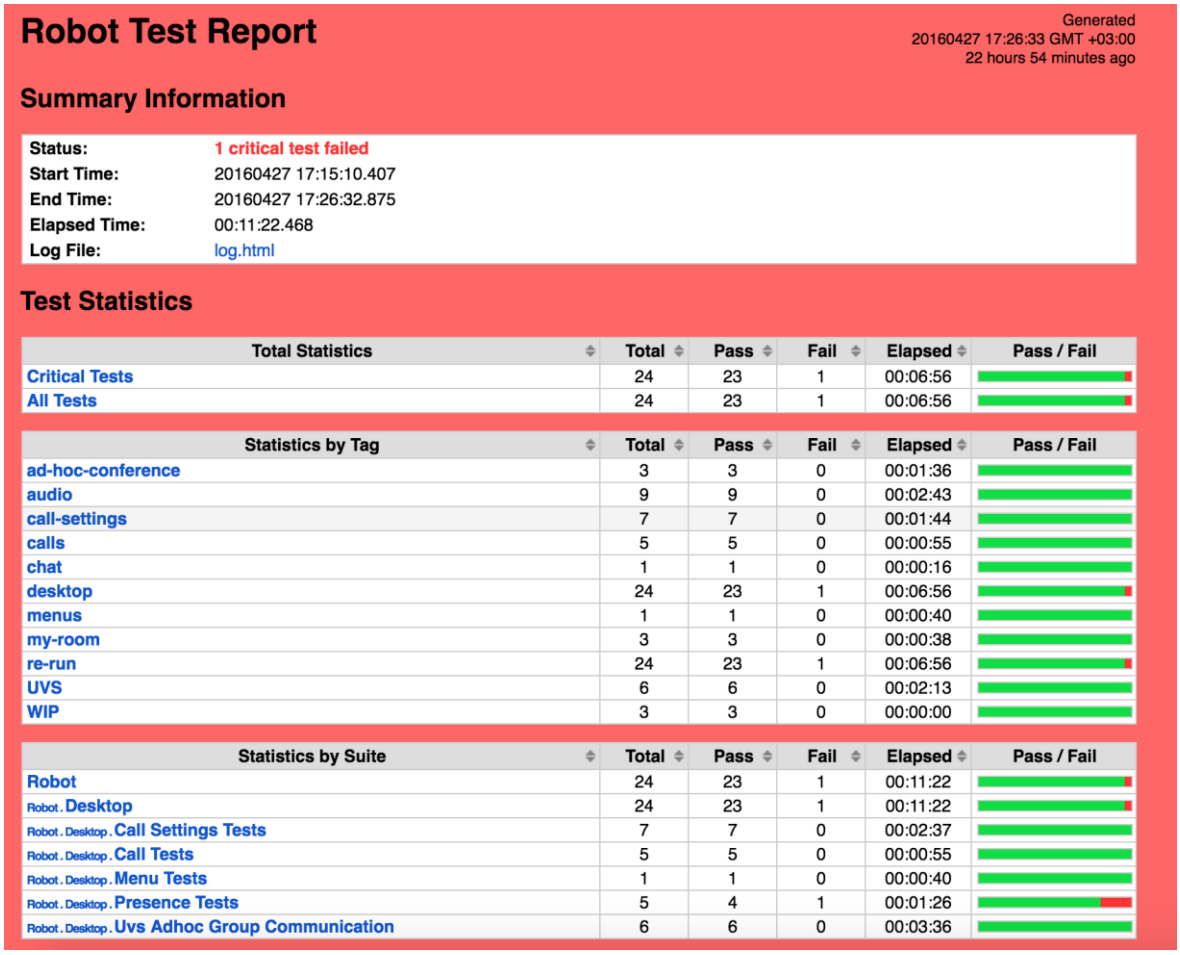
Total Statistics	Total	Pass	Fail	Elapsed	Pass / Fail
Critical Tests	42	42	0	00:09:35	
All Tests	42	42	0	00:09:35	

Statistics by Tag	Total	Pass	Fail	Elapsed	Pass / Fail
Accessibility	1	1	0	00:00:35	
ad-hoc-conference	3	3	0	00:01:28	
audio	11	11	0	00:02:26	
call-settings	7	7	0	00:01:44	
calls	7	7	0	00:00:46	
chat	1	1	0	00:00:15	
configuration-package	11	11	0	00:00:45	
desktop	42	42	0	00:09:35	
menus	17	17	0	00:04:14	
my-room	3	3	0	00:00:36	
UVS	6	6	0	00:02:04	
WIP	3	3	0	00:00:00	

Statistics by Suite	Total	Pass	Fail	Elapsed	Pass / Fail
Robot	42	42	0	00:14:30	
Robot.Desktop	42	42	0	00:14:29	
Robot.Desktop.Audio Only Configuration Tests	11	11	0	00:01:26	
Robot.Desktop.Call Settings Tests	7	7	0	00:02:35	
Robot.Desktop.Call Tests	7	7	0	00:00:46	
Robot.Desktop.Menu Tests	6	6	0	00:03:29	
Robot.Desktop.Presence Tests	5	5	0	00:00:47	
Robot.Desktop.Uvs Adhoc Group Communication	6	6	0	00:03:22	

Test Details

Attachment 5. Screenshot shows summary of robot test cases (Pass/Failure).



Robot Test Report

Generated
20160427 17:26:33 GMT +03:00
22 hours 54 minutes ago

Summary Information

Status:	1 critical test failed
Start Time:	20160427 17:15:10.407
End Time:	20160427 17:26:32.875
Elapsed Time:	00:11:22.468
Log File:	log.html

Test Statistics

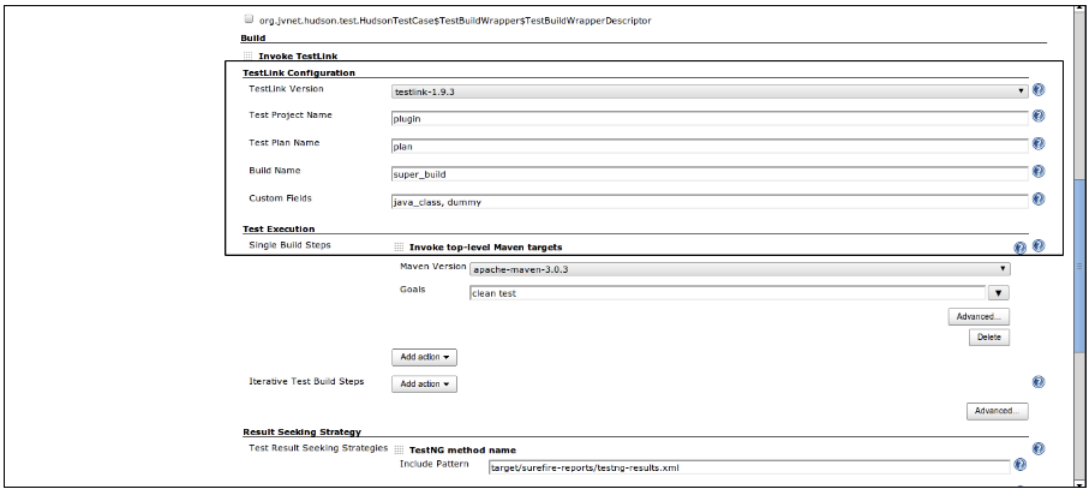
Total Statistics	Total	Pass	Fail	Elapsed	Pass / Fail
Critical Tests	24	23	1	00:06:56	
All Tests	24	23	1	00:06:56	

Statistics by Tag	Total	Pass	Fail	Elapsed	Pass / Fail
ad-hoc-conference	3	3	0	00:01:36	
audio	9	9	0	00:02:43	
call-settings	7	7	0	00:01:44	
calls	5	5	0	00:00:55	
chat	1	1	0	00:00:16	
desktop	24	23	1	00:06:56	
menus	1	1	0	00:00:40	
my-room	3	3	0	00:00:38	
re-run	24	23	1	00:06:56	
UVS	6	6	0	00:02:13	
WIP	3	3	0	00:00:00	

Statistics by Suite	Total	Pass	Fail	Elapsed	Pass / Fail
Robot	24	23	1	00:11:22	
Robot . Desktop	24	23	1	00:11:22	
Robot . Desktop . Call Settings Tests	7	7	0	00:02:37	
Robot . Desktop . Call Tests	5	5	0	00:00:55	
Robot . Desktop . Menu Tests	1	1	0	00:00:40	
Robot . Desktop . Presence Tests	5	4	1	00:01:26	
Robot . Desktop . Uvs Adhoc Group Communication	6	6	0	00:03:36	

Attachment 6. Screenshot show integration of testlink in Jenkins job

TestLink configuration section



Test Execution section